**Subject:** Re: HOT - iOS 12 memory accounting
**From:** "Tammy Levine" <tammy_[redacted]
**Received(Date):** Wed, 20 Feb 2019 01:24:50 +0000
**To:** "Shaan Pruden" <[redacted]
**Cc:** "Allan Schaffer" <[redacted]Trystan Kosmynka" <[redacted]John Geleynse" <[redacted] "Ron Okamoto" <[redacted]CK Haun" <[redacted] "Rob Partington" <[redacted]Michael Wong" <[redacted] "Tom Neumayr" <[redacted]Stephanie Saffer" <[redacted] "Wiley Hodges" <[redacted]
**Date:** Wed, 20 Feb 2019 01:24:50 +0000

---

Thanks for looping us in. When do we expect a decision from ERB on this matter?

On Feb 19, 2019, at 11:57 AM, Shaan Pruden <[redacted] wrote:

+ PR & Wiley

I think they need to weigh in on messaging.

Shaan

> On Feb 19, 2019, at 1:22 AM, Allan Schaffer <[redacted] wrote:
>
> All,
>
> Another aspect of all this is the criticism Unity is taking from their developer community, who are being pretty vocal and perceive this as a clearcut Unity/Apple bug to be fixed.
>
> Unity themselves understand it's not a bug (it's a behavior) and would like to put out some messaging to their users:
>
> If this is the expected behavior we would like to be able to message it appropriately to our users. Specifically something along these lines:
>
> "We have confirmed with Apple that applications running out of memory on iOS 12 (because certain allocations no longer go uncounted) will need to be updated to bring their memory footprint under the limit. The Memory Gauge in Xcode 10.1 should be very helpful in finding the real footprint at runtime. The memory limit available varies from device to device. Devices with higher screen resolutions and less system memory (i.e. older iPads, Apple TV 4th generation, iPhone 6) may be more greatly restricted in available memory then newer devices."
>
> We will then include some recommended tips to users to reduce memory in general. Ideally it would be good include our own calculations to show the degree to which devices are impacted but we would like to get your feedback on this messaging as I could imagine for some

developers this is going to be unwelcome news. Before we proceed with any of this we would certainly want to see that the Radar is reviewed and confirmed not to be an unknown defect.

Seeking your guidance. Should we work with them to craft this support message to put out now, or ask them to hold — pending resolution of the entitlement (or api check) solution and how we'll handle the March deadline?

Allan

On Feb 17, 2019, at 8:08 AM, Trystan Kosmynka <[redacted] wrote:

Thanks John,

This is planned to be an ingest check. I have a note out to Ricardo to see if whitelisting capabilities are built into the check. If they aren't then it's a engineering to do it.

The last time we asked for such an exception it was denied, however there was no customer impact. I think this is different.

On Feb 16, 2019, at 11:28 PM, John Geleynse <[redacted] wrote:

+ Ron
+ Trystan

On Feb 16, 2019, at 4:25 PM, Allan Schaffer <[redacted] wrote:

John, Shaan, ck, wanted to give you a heads-up on an issue that 's been dogging us with Unity, Epic, Aspyr, Codemasters, and many other game developers, that looks to be escalating and ultimately headed to ERB.

**Background**

iOS tracks the memory footprint of running apps, and if the memory footprint for a given app exceeds a certain preset limit it will be terminated. Enforcing these limits allow a big-footprint app to coexist with background apps and other system services (Camera, FaceID, system daemons, etc), and the preset limit varies by device model.

With iOS 11 certain memory allocations (particularly for data shared by the CPU and GPU) were being accounted to the OS kernel instead of being counted toward the limit for the app making the allocations. Metal apps in particular tend to allocate large amounts of GPU resources and these weren't being counted against the limit. Unity's "Corridor" test app, for example, has an actual footprint of 413MB, but only 130MB is accounted to the app and the remaining 283MB went uncounted.

Now in iOS 12 -and if- the developer links with the iOS 12 SDK, memory allocations are correctly accounted to the calling app, allowing iOS to more accurately track the app's total memory footprint. The real amount remains essentially the same — but the portion that was previously counted to the OS kernel is now tallied to the app itself.

The accounting change is activated only if (1) the device is running iOS 12 or later, and (2) the app was linked with the iOS 12 SDK or later. Note that starting March 2019, all iOS apps submitted to the App Store must be built with the iOS 12.1 SDK or later.

**Impact**

With the changed accounting, apps previously operating within the limit for a given device may now run against the limit and be terminated (jetsammed). Indeed several big-name games are finding their memory footprint over the limits now, whereas the same titles worked fine with iOS 11 or when built with a prior SDK version.

Examples of titles impacted:

- Fortnite (Epic)
- Civilization 6 (Aspyr)
- Shadowgun Legends (Madfinger)
- F1 Mobile Racing (Codemasters)

In addition to these titles we hear from Unity they are being hammered via their support channels. Unity is the predominant source of Metal-based apps and games, so they are being hit particularly hard.

**Response**

We have emphasized to Unity & other developers they must update their apps to bring the total memory usage under the limits. This is not being taken well, since it's for a problem the developers perceive to be Apple's own making, and in practice means degrading the experiences for an already-shipping-game on the same devices they're already running on just fine.

Fornite (Epic) in particular has worked closely with the Metal team to bring down their memory footprint. Still they claim their Season 8 is at risk and have requested an exception to allow them to continue building the game with the iOS 11 SDK. With the March 2019 deadline looming, Epic and other developers will be asking for an exception .

Just as of yesterday (02/16), Engineering has proposed to build an entitlement which would let an entitled app revert to the iOS 11-style accounting. This is making its way through the iOS BRB approvals:
<rdar://problem/48129063> Create an entitlement to bypass the correct memory accounting (fixed in iOS 12.0) towards the Jetsam limit

If there is an entitlement, we would (1) need to manage who uses it, (2) still need to provide ERB exceptions until after Peace E ships with it enabled, (3) need to provide messaging or explanation to the rest of the developer community (particularly the Unity-based devs). About (3), Keith Merrill in DTS has started working on a note that could be posted somewhere.

That's where this stands as of today.

This summary is for just us in WWDR, but there's a separate large thread going with engineering about the entitlement. Let me know if you'd like to be added.

Happy to answer any questions as well.

Allan